PATENT ABSTRACTS OF JAPAN

(11)Publication number : 09-006614
(43)Date of publication of application : 10.01.1997

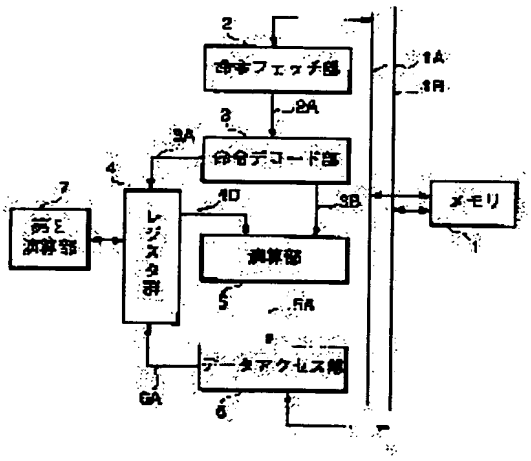
(51)Int.CI. G06F 9/38
G06F 9/30

(21)Application number : 07-154695
(22)Date of filing : 21.06.1995

(71)Applicant : SANYO ELECTRIC CO LTD
(72)Inventor : MIURA HIROKI
KOMURA YASUTO
MATSUMOTO KENJI

(54) DATA PROCESSOR

(57)Abstract:
PURPOSE: To improve performance and to reduce a program size.
CONSTITUTION: An instruction fetch part 2 reads in an instruction and data from a memory 1, and an instruction decoding part 3 decodes them. The operation of a second operation part 7 is started by a data transfer instruction to an input register of a register group 4, and the operation result is stored in an output register. When unprocessed data remains in the input or output register, the succeeding data processing is stopped. Plural sets of input and output registers and computing elements are provided and are selected in accordance with the processing. Since prescribed operation is executed by the data transfer instruction, the processing speed is increased and the program is simplified. The extensibility is high because the addition of a special operation instruction is unnecessary.



LEGAL STATUS

[Date of request for examination] 31.05.2002

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12) 公開特許公報（A）　(11)特許出願公開番号

特開平9－6614

(43)公開日　平成９年(1997)１月10日

(51)Int.Cl.[6]		識別記号	庁内整理番号	ＦＩ			技術表示箇所
Ｇ０６Ｆ	9/38	３７０		Ｇ０６Ｆ	9/38	３７０Ｃ	
	9/30	３７０			9/30	３７０	

審査請求　未請求　請求項の数11　ＯＬ　（全 13 頁）

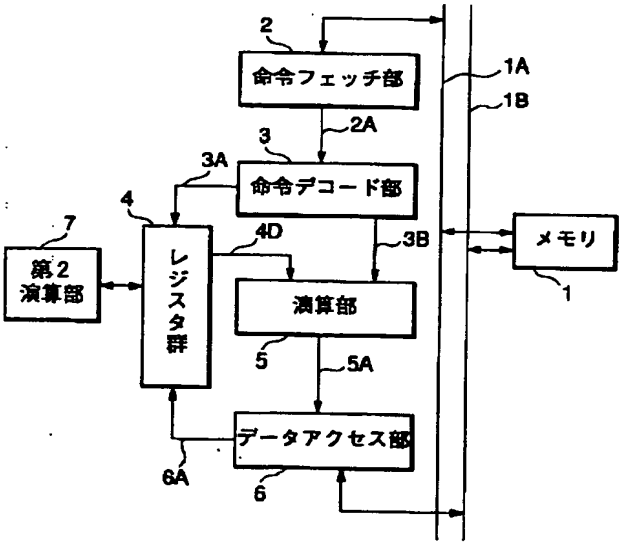
(21)出願番号　特願平7－154695

(22)出願日　平成７年(1995)６月21日

(71)出願人　000001889
三洋電機株式会社
大阪府守口市京阪本通２丁目５番５号
(72)発明者　三浦　宏喜
大阪府守口市京阪本通２丁目５番５号　三洋電機株式会社内
(72)発明者　甲村　康人
大阪府守口市京阪本通２丁目５番５号　三洋電機株式会社内
(72)発明者　松本　健志
大阪府守口市京阪本通２丁目５番５号　三洋電機株式会社内
(74)代理人　弁理士　吉田　研二　（外２名）

(54)【発明の名称】　データ処理装置

(57)【要約】
【目的】　性能向上とプログラムサイズの縮小。
【構成】　命令フェッチ部２がメモリ１から命令とデータを読み込み、命令デコード部３で解読する。レジスタ群４の入力用レジスタへのデータ転送命令によって第２演算部７の演算が起動され、演算結果は出力用レジスタに格納される。入力用または出力用レジスタに未処理のデータが残っていれば、後続のデータ処理を停止する。入力用、出力用レジスタセット、演算器を複数設け、処理に応じて択する。データ転送命令で所期の演算が実行されるため、処理の高速化、プログラムの簡素化が可能となる。特別な演算命令の追加が不要のため、拡張性も高い。



1

【特許請求の範囲】
【請求項１】　各種命令を実行してデータ処理を行うデータ処理装置であって、
演算命令の実行によって所定の演算処理を起動する第１演算手段と、
所定の入力用レジスタ及び出力用レジスタを含み、データ転送命令によるアクセスが可能な複数のレジスタと、
データ転送命令の実行によって前記入力用レジスタに対するデータ転送が行われたことを契機としてそのデータに対する所定の演算処理を起動し、その演算結果データを前記出力用レジスタに書き込む第２演算手段と、
前記演算結果データの読み出しが行われるまで、前記出力用レジスタに対する後続の演算結果データの書き込みを待たせる書込停止手段と、
を備えたことを特徴とするデータ処理装置。
【請求項２】　請求項１に記載のデータ処理装置において、該装置はさらに、
命令の実行が前記演算結果データの読み出しを伴うとき、所望の演算結果データが前記出力用レジスタに書き込まれるまで読み出しを待たせる読出停止手段を含むことを特徴とするデータ処理装置。
【請求項３】　請求項１に記載のデータ処理装置において、該装置はさらに、
前記書込停止手段によって書き込みが待たされているときに、前記入力用レジスタに対するデータ転送が行われた場合、前記第２演算手段による演算処理の起動を待たせる演算停止手段を含むことを特徴とするデータ処理装置。
【請求項４】　請求項１に記載のデータ処理装置において、該装置はさらに、
前記入力用レジスタに有効なデータが保持されているとき、後続のデータ転送命令の実行による該レジスタへのデータ転送を待たせる転送停止手段を含むことを特徴とするデータ処理装置。
【請求項５】　請求項１に記載のデータ処理装置において、
該装置は、前記入力用レジスタ及び出力用レジスタの組合せによる複数のレジスタセットを備え、さらにこれら複数のレジスタセットから演算処理を行うべきレジスタセットを指定する指定手段を有することを特徴とするデータ処理装置。
【請求項６】　請求項１に記載のデータ処理装置において、
前記第２演算手段は、それぞれが異なる演算を行う複数の演算器を有し、
該装置は、これら複数の演算器のうちデータ転送が行われたとき実際に演算処理を起動すべきものを指定する指定手段を有することを特徴とするデータ処理装置。
【請求項７】　請求項６に記載のデータ処理装置において、

2

前記指定手段は、前記入力用レジスタおよび出力用レジスタとは別に設けられた制御レジスタであることを特徴とするデータ処理装置。
【請求項８】　請求項１に記載のデータ処理装置において、
前記第２演算手段は、それぞれが異なる演算を行う複数の演算器を有し、
該装置は、
前記複数の演算器のうち１つの演算器の演算結果データを別の演算器へ入力する手段と、
前記出力用レジスタに書き込まれたデータを前記別の演算器に入力する手段と、
前記別の演算器の演算結果データを前記出力用レジスタに再帰的に入力する手段と、
を有することを特徴とするデータ処理装置。
【請求項９】　請求項１に記載のデータ処理装置において、
該装置は、前記入力用レジスタを複数備え、
これらの入力用レジスタに対して順次データ転送がなされるとき、１回の演算処理に必要な複数のデータが入力用レジスタに揃うタイミングを検出する検出手段と、
これらのデータが揃うまで、前記第２演算手段による演算処理の起動を待たせる演算停止手段を含むことを特徴とするデータ処理装置。
【請求項１０】　請求項８に記載のデータ処理装置において、
前記入力用レジスタと前記第２演算手段との間に介挿され、前記入力用レジスタに転送されたデータを順次記憶するＦＩＦＯメモリと、
前記ＦＩＦＯメモリからのデータ読み出しを制御する制御手段と、
を有することを特徴とするデータ処理装置。
【請求項１１】　請求項９に記載のデータ処理装置において、
前記制御手段は、演算処理に必要な複数のデータが前記ＦＩＦＯに揃ったとき前記ＦＩＦＯメモリからのデータ読み出しを行い、
前記演算停止手段は、前記ＦＩＦＯメモリからデータが読み出されたとき前記第２演算手段による演算処理の起動を許可することを特徴とするデータ処理装置。
【発明の詳細な説明】
【０００１】
【産業上の利用分野】本発明は、各種命令を実行してデータ処理を行うデータ処理装置に関する。
【０００２】
【従来の技術】マイクロプロセッサを代表とするデータ処理装置として、メモリに格納された命令を順次フェッチ、デコードして実行する制御駆動型の装置が数多く存在する。
【０００３】図１はＲＩＳＣ（reduced instruction se

t computer）方式のプロセッサの構成図である。このプロセッサは、メモリ１、命令フェッチ部２、命令デコード部３、レジスタ群４、演算部５、データアクセス部６を持つ。この構成において、メモリ１から順次フェッチされる演算命令、データ転送（ロード／ストア）命令などがデコードされ、演算部５において、命令コードで指示される演算が実施される。

【０００４】従来、図１の演算部５の中に、加算、減算、論理和、論理積などの演算を実行する基本的な算術論理演算部（ＡＬＵ）と別に、例えば乗算を高速に実行するための乗算器を並列的に備えるものが存在した。また、整数演算用のＡＬＵと浮動小数点演算用のＡＬＵが並設されたものもあった。演算部５を構成する各種演算器を利用してデータ処理を行う場合、演算器による演算処理に対応する演算命令を設け、これをプログラム中に記述してプロセッサに読み込む方法が一般的だった。すなわち、例えばＡＤＤ（整数加算）、ＳＵＢ（整数減

```
LD    R2, (R0)
LD    R3, (R1)
MUL   R2, R3
```

この結果、以下の課題があった。

【０００７】［課題１］処理性能の限界

仮に、１命令サイクルで演算を実行する高速の演算器を搭載しても、全体の処理レートとしては、３命令サイクルでようやく１回の演算にとどまる。これが性能向上の足かせとなる。

【０００８】［課題２］プログラムサイズの増大

演算処理に必要な命令数が多いため、プログラムサイズ全体が大きくなる。

【０００９】［課題３］命令コードの長大化

演算処理を新規に追加するとき、命令コード中に処理内容を示す演算コードを追加して割り当てる必要があるため、処理機能の拡充が命令コード長の長大化に直結する。この結果、課題２が顕著になり、プロセッサのハードウェア規模も大きくなる。

【００１０】［課題４］命令のコンパクト設計の困難性

基本演算以外の演算処理も将来的に追加、拡張されうるため、予め仕様に組み込む必要がある。このため、多くの演算コードの割当てによって命令コード長のコンパクトな設計が困難となる。

【００１１】［課題５］拡張性を考慮したプロセッサ設計の困難性

長期的な演算処理の追加・拡張をすべて考慮した設計自体極めて困難であり、追加等は自ずと限定される。柔軟かつ拡張性を保持したプロセッサの設計は非常に困難である。

【００１２】これらの課題を一部解決する技術として、日本電気（株）のＤＳＰであるμＰＤ７７２４０（商標）を挙げることができる。「ユーザーズ・マニュアル　μＰＤ７７２４０」（１９９１年９月版）の６６ページの記載によれば、このＤＳＰの特徴は以下の通りである。

【００１３】（１）通常のＡＬＵの他に浮動小数点乗算専用の回路（ＦＭＰＹ）を持つ。前者は明示的な演算命令（ＡＤＤ、ＳＵＢなど）によって演算を起動し、後者はデータ転送命令によって、転送されたデータに対する演算を自動的に起動する。

【００１４】（２）ＦＭＰＹでは、乗算入力用レジスタＫ、Ｌの２つのレジスタに転送されるデータが命令サイクルごとに掛け合わされ、乗算結果が１命令サイクル後にＦＭＰＹの出力バスに出力され、２命令サイクル後に乗算出力用レジスタＭに書き込まれる。

【００１５】この結果、データ転送命令のみの記述によって乗算起動が可能となり、上記課題１及び２にある程度の解決を与えるものである。

【００１６】

【発明が解決しようとする課題】しかしこのＤＳＰでも、以下の問題が残る。

【００１７】１．課題１、２について

図２はこのＤＳＰの特定命令のフィールド構成図である。この構成は上記マニュアルの１４ページに掲載される「オペレーション命令」と総称される命令のフィールド構成で、その概要は以下通りである。

【００１８】１．命令コード長は３２ビットである

２．ＯＰは演算処理の種類を示す

３．ＣＮＴはＤＳＰの内部状態の変更指示、例えばアドレスレジスタのインクリメント等を行う

４．Ｑ、Ｐはそれぞれ第１、第２オペランドを表す

５．ＳＲＣ、ＤＳＴはそれぞれ転送命令の転送元、転送先を示す

すなわち、原則として３２ビット固定の命令の中に、演算、転送、内部状態指示という、性質の異なる命令（以下「個別命令」という）が３つ並記される（同図下を参照）。これは、ＤＳＰの多くがマイクロプログラム制御計算機のアーキテクチャを踏襲するためであり、結果的に、ある程度命令コード長が長くなることと引き換えに、合計ステップ数を削減する効果を持つ。この際、各個別命令の記述に必要なビット数の多寡をうまく組み合わせることによって、これらを３２ビットに収めるため、このアーキテクチャは課題１、２を解決する方向にある。

【００１９】しかし、現実のプログラミングを考えると、この技術による改善効果はさして大きくない。すなわち、効率の最大化を図る場合、常に３つの並記可能な個別命令をセットにする必要がある一方、実際にはそうした配列が不可能だからである。例えば、転送命令のみを数回続けなければならない場合、実質的に上記ＳＲＣ、ＤＳＴのフィールドのみ（計１０ビット）を記述するために、毎回３２ビットの命令コードが必要となり、この間メモリの使用効率は約１／3となる。この結果、再度課題１、２が問題として現れる。

【００２０】２．課題３～５について

このＤＳＰでは、ＦＭＰＹによる演算が乗算に限られている。しかし将来別の演算が追加される場合、なんらかの方法で、複数の演算から所期の演算を選択する必要が生じる。図２の構成を採る限り、こうした選択はＯＰに新たな個別命令を追加することによって行われるが、こうした対処は、データ転送に関する個別命令のみによって乗算が実行されるというＦＭＰＹの設計思想に沿わないだけでなく、課題３～５に直結するおそれがある。もともと、このＤＳＰはこれらの課題を念頭において設計されたものではない点に注意すべきである。

【００２１】以上が課題１～５とＤＳＰの関係である。なお、本発明の主用途であるＲＩＣＳプロセッサを考えた場合、このＤＳＰにはさらに次の課題が考えられる。

【００２２】［課題６］プログラミングの困難性

ＦＭＰＹによれば、常時Ｋ、Ｌレジスタに対する乗算が行われており、２サイクル前の計算結果がＭレジスタに現れる。計算結果は参照の有無に関係なく１サイクル毎に更新されるため、アセンブリ言語を使用するプログラマは、Ｍレジスタから所望の計算結果を読み出す際、タイミングに細心の注意を払わねばならない。一方、高級言語を使用する場合は、プラグラマの意図した読み出しタイミングを実現するコンパイラの開発が極めて困難である。

【００２３】また、仮にタイミングの問題を解決したとしても、計算結果の読み出し前に割込みが発生すれば別の問題が発生する。割込み処理が終了するまでに、必要な計算結果が失われうるためである。乗算前に割込み禁止を設定する方法もあるが、（１）プログラムが複雑になり、（２）割込みの禁止許可はシステムレベルで判断すべき問題であり、（３）ＦＭＰＹの目的である即時性が失われる、の諸点から現実的でない。

【００２４】［課題７］アーキテクチャに起因する制限

このＤＳＰでは、Ｋ、Ｌ、Ｍレジスタセットを１つしか持たず、これとＦＭＰＹが組として設けられているため、仮に新たな演算（例えば加算）を追加する場合、それに応じて別のレジスタセット（例えばK_2、L_2、M_2）を追加しなければならない。ＲＩＣＳプロセッサでは、画像処理やＣＧなどの科学技術計算を実施するために、加減乗除、三角関数等各種演算が必要であり、演算ごとに３２ビットレジスタのセットを増設していくとすればハードウエアの大規模化を招く。

【００２５】［課題８］ハードウエアの規模

このＤＳＰでは、Ｋ、Ｌレジスタに同時にデータを転送することによって高速化を図っている。このため、これらのレジスタの入り口に二重構造の３２ビットデータバスを採用する。従ってこの部分のハードウエアが複雑であり、回路規模も大きい。コンパクト設計、製品コストの観点からは改善することが望ましい。

【００２６】本発明の目的

本発明の目的は、従来一般的に問題となった課題１～５のみならず、上記したＤＳＰの課題６～８をも解決するデータ処理装置を提供することにある。

【００２７】

【課題を解決するための手段】本発明のデータ処理装置は、演算命令の実行によって所定の演算処理を起動する第１演算手段と、所定の入力用レジスタ及び出力用レジスタを含み、データ転送命令によるアクセスが可能な複数のレジスタと、データ転送命令の実行によって前記入力用レジスタに対するデータ転送が行われたことを契機としてそのデータに対する所定の演算処理を起動し、その演算結果データを前記出力用レジスタに書き込む第２演算手段と、前記演算結果データの読み出しが行われるまで、前記出力用レジスタに対する後続の演算結果データの書き込みを待たせる書込停止手段とを備える。

【００２８】本発明は、命令の実行が前記演算結果データの読み出しを伴うとき、所望の演算結果データが前記出力用レジスタに書き込まれるまで読み出しを待たせる読出停止手段を含む。

【００２９】本発明は、前記書込停止手段によって書き込みが待たされているときに、前記入力用レジスタに対するデータ転送が行われた場合、前記第２演算手段による演算処理の起動を待たせる演算停止手段を含む。

【００３０】本発明は、前記入力用レジスタに有効なデータが保持されているとき、後続のデータ転送命令の実行による該レジスタへのデータ転送を待たせる転送停止手段を含む。

【００３１】本発明は、前記入力用レジスタ及び出力用レジスタの組合せによる複数のレジスタセットを備え、

さらにこれら複数のレジスタセットから演算処理を行うべきレジスタセットを指定する指定手段を有する。

【００３２】前記第２演算手段は、それぞれが異なる演算を行う複数の演算器を有し、本発明の装置は、これら複数の演算器のうちデータ転送が行われたとき実際に演算処理を起動すべきものを指定する指定手段を有する。

【００３３】前記指定手段は、前記入力用レジスタおよび出力用レジスタとは別に設けられた制御レジスタである。

【００３４】前記第２演算手段は、それぞれが異なる演算を行う複数の演算器を有し、本発明の装置は、前記複数の演算器のうち１つの演算器の演算結果データを別の演算器へ入力する手段と、前記出力用レジスタに書き込まれたデータを前記別の演算器に入力する手段と、前記別の演算器の演算結果データを前記出力用レジスタに再帰的に入力する手段とを有する。

【００３５】本発明は、前記入力用レジスタを複数備え、これらの入力用レジスタに対して順次データ転送がなされるとき、１回の演算処理に必要な複数のデータが入力用レジスタに揃うタイミングを検出する検出手段と、これらのデータが揃うまで、前記第２演算手段による演算処理の起動を待たせる演算停止手段を含む。

【００３６】本発明は、前記入力用レジスタと前記第２演算手段との間に介挿され、前記入力用レジスタに転送されたデータを順次記憶するＦＩＦＯメモリと、前記ＦＩＦＯメモリからのデータ読み出しを制御する制御手段とを有する。

【００３７】

【作用】本発明のデータ処理装置によれば、第１演算手段において演算命令の実行により、所定の演算処理が起動される。一方、データ転送命令の実行によって入力用レジスタに対するデータ転送が行われたとき、第２演算手段においてそのデータに対する所定の演算処理が起動され、その演算結果データが出力用レジスタに書き込まれる。この際、演算結果データの読み出しが行われるまで、出力用レジスタに対する後続の演算結果データの書き込みが待たされる。

【００３８】別の態様では、命令の実行が前記演算結果データの読み出しを伴うとき、所望の演算結果データが前記出力用レジスタに書き込まれるまで読み出しが待たされる。

【００３９】また別の態様では、前記書き込みが待たされているときに、前記入力用レジスタに対するデータ転送が行われた場合、前記第２演算手段による演算処理の起動が待たされる。

【００４０】さらに別の態様では、前記入力用レジスタに有効なデータが保持されているとき、後続のデータ転送命令の実行による該レジスタへのデータ転送が待たされる。

【００４１】前記入力用レジスタ及び出力用レジスタの組合せによるレジスタセットが複数存在する場合は、これら複数のレジスタセットから演算処理を行うべきレジスタセットが指定される。

【００４２】前記第２演算手段がそれぞれ異なる演算を行う複数の演算器を有する場合は、これら複数の演算器のうちデータ転送が行われたとき実際に演算処理を起動すべきものが指定される。

【００４３】この指定は、前記入力用レジスタおよび出力用レジスタとは別に設けられた制御レジスタによって行われる。

【００４４】この場合、別の態様によれば、前記複数の演算器のうち１つの演算器の演算結果データが別の演算器へ入力され、前記出力用レジスタに書き込まれたデータが前記別の演算器に入力され、前記別の演算器の演算結果データが前記出力用レジスタに再帰的に入力される。

【００４５】前記入力用レジスタが複数存在する場合、これらの入力用レジスタに対して順次データ転送がなされるとき、１回の演算処理に必要な複数のデータが入力用レジスタに揃うまで、前記第２演算手段による演算処理の起動が待たされる。

【００４６】前記入力用レジスタと前記第２演算手段との間にＦＩＦＯメモリが介挿される場合は、このメモリが前記入力用レジスタに転送されたデータを順次記憶する。記憶されたデータは、必要なタイミングで順次読み出される。

【００４７】より具体的には、演算処理に必要な複数のデータが前記ＦＩＦＯに揃ったとき前記ＦＩＦＯメモリからデータの読み出しが行われ、前記ＦＩＦＯメモリからデータが読み出されたとき、前記第２演算手段による演算処理の起動が許可される。

【００４８】

【実施例】以下、本発明の好適な実施例を適宜図面を参照しながら説明する。

【００４９】実施例１．本発明のデータ処理装置では、図１同様の演算部における演算が演算命令によって起動される一方、新たな演算部（以下、第２演算部という）における演算処理が特定のレジスタに対するデータ転送命令のみによって起動される。

【００５０】図３は本実施例に係るデータ処理装置の全体構成図である。

【００５１】同図において、メモリ１には命令およびデータが格納される。命令は命令バス１Ａ、データはデータバス１Ｂを介してアクセスされる。命令フェッチ部２はメモリ１からデータ転送命令や演算命令などの命令を順次フェッチし、信号線２Ａを介してこれを命令デコード部３に渡す。

【００５２】命令デコード部３では命令の種類に従って命令コードを所定のフィールドに分割し、処理すべき演算の種類を示すオペレーション、命令コード内に埋め込

まれたオペランドである即値オペランド、ソースレジスタ番号、ディスティネーションレジスタ番号などを抽出する。オペレーションは信号線３Ｂを介して演算部５へ送られ、レジスタ群４は信号線３Ａを介してアクセスされる。

【００５３】次にレジスタ群４からソースレジスタ番号で指定されたレジスタの内容が読み出され、信号線４Ｄを介して演算部５へ送られる。オペレーションが加減算などの演算命令を示していれば、ソースオペランドに対し、演算部５で通常の演算が行われる。演算結果は信号線５Ａを介してデータアクセス部６へ送られ、信号線６Ａを介してレジスタ群４中のディスティネーションレジスタ番号で示されるレジスタに書き戻される。

【００５４】一方、オペレーションがメモリへのストア命令なら、演算部５によって生成されるデータアドレスと書き込みデータにより、データバス１Ｂを介してメモリ１にデータが書き込まれる。また、ロード命令なら、演算部５によって生成されるデータアドレスに従ってメモリ１の内容が読み出され、レジスタ群４中の指定されたレジスタに書き込まれる。

【００５５】以上の構成は従来の装置とほぼ同様であるが、本実施例の特徴はレジスタ群４に接続された第２演算部７にある。

【００５６】図４は図３のレジスタ群４と第２演算部７の接続を示す図である。同図において、Ｒ０～３およびＳＲ０～２は作業用レジスタであり、ロード命令、ストア命令などのデータ転送命令によってアクセスされる。Ｒ０～３は汎用の作業用レジスタ、ＳＲ０～２は第２演算部７による使用が可能なデータ格納用レジスタである。ＳＲ０、１、２はそれぞれデータ線４Ａ、４Ｂ、７Ａによって第２演算部７と接続される。さらにＳＲ２から信号線４Ｃにより、後述の出力データ存在フラグの状態が第２演算部７へ送られる。本実施例では、ＳＲ０、１を第２演算部７への入力データ格納用、ＳＲ２を第２演算部７からの出力データ格納用に用いる。

【００５７】図５は第２演算部７の内部構成図である。同図に示す通り、第２演算部７は実際の演算を行う演算回路７１と、これを制御する制御回路７２からなる。演算回路７１は２入力１出力であり、例えば乗算、除算などを行う。同図中、入力データ存在フラグ０、１はそれぞれ処理の対象となるデータがＳＲ０、１に存在するか否かを示し、存在するときにセット状態、しないときにリセット状態になる。これらフラグの状態は信号線４Ａｆ、４Ｂｆを介して制御回路７２に与えられる。

【００５８】ＳＲ０、１から読み出されたデータ自体は演算回路７１に与えられる。演算回路７１に有効な演算結果が生じたとき、制御回路７２は演算結果存在フラグをセットし、信号線７Ａｅに出力する。演算回路７１は演算結果データをデータ線７Ａｄに出力する。信号線７Ａｅとデータ線７Ａｄによってデータ線７Ａが構成され、これらがＳＲ２へ与えられる。なお、制御回路７２は制御線７２Ａにより、演算回路７１に対して演算の起動を許可する。

【００５９】以下、図３～５を用いて本実施例に特徴的なデータ処理動作を説明する。

【００６０】１．ＳＲ０へのデータロード

まず「メモリからレジスタＳＲ０へのロード」命令がフェッチされたとする。このときメモリから必要なデータが読み出され、データ線６Ａを経てＳＲ０にロードされ、前述の入力データ存在フラグ０がセットされる。データは演算回路７１に与えられる。

【００６１】２．ＳＲ１へのデータロード

つづいて、「メモリからレジスタＳＲ１へのロード」命令がフェッチされると、同様にメモリからデータが読み出され、ＳＲ１にロードされる。ここで入力データ存在フラグ１もセットされ、データは演算回路７１へ与えられる。

【００６２】３．演算の起動

制御回路７２は、入力データ存在フラグ０、１がともにセット状態であることを確認し、制御線７２Ａを介して演算起動指令を演算回路７１に与える。演算回路７１は指令に応じて演算を起動、実行する。

【００６３】４．ＳＲ２に対する演算結果の格納

制御回路７２は演算回路７１における演算実行に必要な時間の経過後、演算結果存在フラグをセットする。演算結果データとこのフラグはデータ線７Ａを介してレジスタＳＲ２に与えられる。

【００６４】ＳＲ２は、出力データ存在フラグ、すなわち自レジスタに有効な演算結果データが存在するか否かを示すフラグを持つ。このフラグは初期状態でリセットされており、かつ、リセット状態にある場合に限って演算結果の書き込みが許可される。書き込み後、このフラグはセットされる。出力データ存在フラグがセットされたとき、演算結果存在フラグはリセットされる。

【００６５】５．ＳＲ２からメモリへのデータストア

この後、「ＳＲ２からメモリへのストア命令」がフェッチされると、データ線４Ｄを介してＳＲ２の読み出しが行われ、出力データ存在フラグは再度リセットされる。

【００６６】以上が一連の処理の例である。以降、制御回路７２は同様に、入力データ存在フラグ０、１がともにセット状態にあることを検出するたびに、演算回路７１に演算起動指令を送り、演算が完了すれば演算結果存在フラグをセットする。ただし、演算結果存在フラグがセットされたとき、出力データ存在フラグがいまだセット状態にあれば、ＳＲ２への演算結果の書き込みを停止する。

【００６７】さらに、演算結果存在フラグがセット状態にあり、かつＳＲ２への書き込みも停止されていれば、この間制御回路７２は演算回路７１に対する演算起動指令の発行を停止する。

【００６８】これらの配慮の結果、プログラマまたはコンパイラは所期データの読み出しタイミングに注意を払う必要がなくなる。すなわち本実施例によれば、必要なデータが必要な箇所に準備されたことを確認して、はじめて後続の処理が行われるため、いかなるタイミングで演算結果の読み出し命令を実行しても、いかなるタイミングで入力データをセットする命令を実行しても、正しい処理を行うことができる。この結果、従来の課題６を解決するものである。

【００６９】本実施例では、演算部５における演算が従来通り演算命令によって起動される一方、第２演算部７における演算処理はＳＲ０、１へのデータ転送命令のみによって起動される。演算回路７１が乗算器のとき、メモリに格納された各々異なるオペランドデータに対する乗算を繰り返し実行する場合、例えば下記のプログラムとなる。ここでは１回の乗算あたり２命令しか必要とせず、処理の高速化とプログラムサイズの縮小が実現される。

【００７０】

```
LD  SR0, (R0)   ( SR0 ← men(R0) )
LD  SR1, (R1)   ( SR1 ← men(R1) )    乗算起動
```

また、別の処理として、メモリに格納された一次元配列データ（すなわちデータストリーム）どうしの積を計算し、その結果の一次元配列をメモリ格納する場合を考える。こうした処理は、音声処理や画像処理の応用分野で頻繁に実行されるものである。本実施例では、乗算に第２演算部７、配列アドレスの計算に演算部５を使う。プログラム例を示す。

【００７１】

```
( SR0 ← men(R0), R0  ← R0+1 )
( SR1 ← men(R1) )  乗算起動
  加算と乗算の並列処理
( mem(R2) ← SR2, R2  ← R2+1 )
```

```
LD   SR0, (R0++)
LD   SR1, (R1)
ADD  R1, 8 (R1←R1＋8)
ST   SR2, (R2++)
```

ここで３行目の数値「８」は、ポインタＲ１に与えるべきシフト量である。このプログラムの場合、２行目のロード命令の実行によって第２演算部７の乗算が起動され、３行目の加算命令の実行によって演算部５の加算が起動され、加算と乗算が同時並列的に処理される。この後、ストア命令により、ＳＲ２に書き込まれた乗算結果がメモリにストアされる。このプログラムを従来型のＲＩＳＣプロセッサで実行する場合、乗算命令（ＭＵＬ）が必要となり、処理速度とプログラムサイズの面で不利である。

【００７２】以上、これらのプログラムから明らかなように、本実施例によれば従来の課題６のみならず、課題１、２をも解決することができる。

【００７３】本実施例では、第２演算部７が２入力であり、２つのデータがＳＲ０、１に書き込まれたことを検出して演算を起動したが、例えばＳＲ０にデータが書き込まれた時には入力データ存在フラグ０を出さず、ＳＲ１にデータが書き込まれた時に入力データ存在フラグ０、１を同時に出力する構成としてもよい。なお、この演算の場合、例えば定数データをＳＲ０に格納しておけば、ＳＲ１へのロード命令のみを連続実行することにより、演算を繰り返し行うことができる。これらの構成によれば、従来の課題８をも同時に解決することができる。

【００７４】実施例２．つぎに、実施例１の改良例である実施例２を説明する。実施例１の構成では、例えばＳＲ０、１へのロード命令が連続的に実行されたとき、ＳＲ２の読み出しがボトルネックとなり、演算処理全体が停止する場合がある。本実施例はこうした性能低下を回避するものである。

【００７５】図６は実施例２の第２演算部７の内部構成図、図７は図６の制御回路７２の内部構成図である。これら以外の部分は実施例１と同等でよく、ここでは相違点のみを説明する。

【００７６】本実施例の特徴は、演算回路７１の２つの入力部に、ＦＩＦＯ（先入れ先出し）メモリ７３、７４を設け、それぞれＳＲ０、１への連続的書き込みを可能にする点にある。図６に示すごとく、ＦＩＦＯメモリ７３、７４は制御回路７１から書き込み信号７２１、７２２、および読み出し信号７２３によって制御される。一方、図７に示す通り、制御回路７２は以下の構成を持つ。

【００７７】（１）計数回路Ａ７２４、Ｂ７２７
それぞれＦＩＦＯメモリ７３、７４内に記憶されているデータ数を計数する。これらの回路の初期出力値は０で、それぞれ入力データ存在フラグ０、１がセットされるたびに信号線４Ａｆ、４Ｂｆによってカウントアップされ、カウントダウン信号７３１によってカウントダウンされる。

【００７８】（２）入力制御回路Ａ７２５
入力データ存在フラグ０がセットされたとき、計数回路Ａ７２４の示すデータ量がＦＩＦＯメモリ７３の最大容量未満であれば、そのメモリに対する書き込み信号７２１を出力し、ＳＲ０から送られたデータを書き込む。また、計数回路Ａ７２４の示すデータ量が０でない限り、常に読み出し要求信号７２６を出力する。

【００７９】（３）入力制御回路Ｂ７２８
入力制御回路Ａ７２５同様の制御をＦＩＦＯメモリ７４について行う。すなわち、入力データ存在フラグ１がセットされたとき、計数回路Ｂ７２７の示すデータ量がＦ

ＩＦＯメモリ７４の最大容量未満なら書き込み信号７２２を出力し、ＳＲ１から送られたデータを書き込む。また、計数回路Ｂ７２７の示すデータ量が０でない限り、常に読み出し要求信号７２９を出力する。

【００８０】（４）演算制御回路７３０

読み出し要求信号７２６、７２９の両方が出力されたとき、演算対象である２つのデータがレジスタに揃ったと判断する。データが揃い、かつ図６の演算結果存在フラグがリセット状態にあるとき、読み出し信号７２３をＦＩＦＯメモリ７３、７４に対して同時に出力し、これらのメモリから２つのデータを読み出す。読み出されたデータは演算回路７１に与えられる。このとき同時に、制御線７２Ａを介して演算回路７１に演算の起動を許可する。さらに、計数回路Ａ７２４、Ｂ７２７に向けてカウントダウン信号７３１を出力する。

【００８１】以上が実施例２の概要である。この構成によれば、データの入力を連続して行うことができる。従って、実施例１の構成による性能の低下を回避することができ、従来の課題６を解決しつつ課題１に対して別の方法による解決を与えるものである。またこの際、ＳＲ０、１に対してデータを同時に入力する必要もなく、課題８も解決される。例えば、実施例１で述べた一次元配列どうしの積和計算を考える場合、２つのデータストリームを別々のレジスタに向けて非同期的に投入し、所定のレジスタから非同期的に読み出すようなプログラミングが許され、かつその処理時間も短い。こうした特徴から、本実施例は複雑な科学計算などで大きな効果を発揮する。

【００８２】実施例３．つづいて、実施例１の別の改良例である実施例３を説明する。

【００８３】図８は本実施例に係るデータ処理装置の全体構成図、図９は図８のレジスタ群４と第２演算部７の接続を示す図である。これらの図はそれぞれ、実施例１の図３、４に対応する。以下、実施例１と同じ構成と動作については説明を省略する。

【００８４】実施例３で新設される構成は、

（１）出力データ存在フラグの状態をデータアクセス部６に伝える信号線４Ｃ’

（２）入力データ存在フラグ０、１の状態を命令デコード部３へ伝える信号線４Ｊ（信号線４Ｊは信号線４Ａｆと４Ｂｆの総括名称）

（３）本装置によるパイプライン処理動作に対する停止要求信号３Ｃおよび６Ｂ

（４）停止要求信号３Ｃ、６Ｂのワイヤード・オアをとる、例えばオープンドレインタイプのゲートｇ１、ｇ２

（５）前記ワイヤード・オアの結果生じる停止信号１００

である。この構成による動作の特徴は次の通りである。

【００８５】１．ＳＲ２からの読み出し命令実行の際、出力データ存在フラグがセットされていなければ、データアクセス部６が停止要求信号６Ｂを出力し、ゲートｇ２を経て停止信号１００が出力される。この結果、装置は実行待ち状態になる。

【００８６】２．ＳＲ０にデータを書き込む命令を実行する際に入力データ存在フラグ０がセット状態にあるとき、またはＳＲ１にデータを書き込む命令を実行する際に入力データ存在フラグ１がセット状態にあるとき、命令デコード部３が停止要求信号３Ｃを出力し、ゲートｇ１を経て停止信号１００が出力される。この結果、装置は実行待ち状態になる。

【００８７】以上が実施例３の概要である。本実施例によれば、必要に応じて装置のパイプライン処理自体が停止されるため、演算結果の読み出しタイミングに従来のような制約がない。この結果、前述の課題６を根本的に解決することができる。なお、本実施例においても実施例２同様の配慮を行うことにより、性能の低下を回避することが望ましい。

【００８８】実施例４．つづいて実施例４を説明する。図１０は本実施例のレジスタ群４と第２演算部７の接続を示す図である。その他の構成はこれまでの実施例と同等である。

【００８９】実施例４では、第２演算部７のために使用できる作業用レジスタとして、ＳＲ０～５の６本のレジスタを備え、レジスタセットＳＲ０～２またはＳＲ３～５のいずれか一方が選択的に使用されるものとする。レジスタセットの選択は、新たな構成である選択回路４１、４２とコントロールレジスタ（ＣＴＬＲ）による。選択回路４１、４２の実体は、例えば２入力１出力のセレクタ群である。選択回路４１、４２による選択は、コントロールレジスタのｓｅｌビットによる。レジスタセットが３以上存在するときは、複数のｓｅｌビット（ｓｅｌフィールド）によって選択を行う。

【００９０】この構成より、予めコントロールレジスタによって選択すべきレジスタセットを設定すると、コントロールレジスタから制御信号４Ｅが選択回路４１、４２に与えられる。以降、選択されたレジスタセットにより、実施例１同様の処理がなされる。

【００９１】以上、本実施例によれば、例えば割込み処理等の実行に通常とは別のレジスタセットを使用することができ、割込み処理中に所望の演算結果が失われるおそれがない。従って、従来の課題７を解決することができる。

【００９２】実施例５．図１１は実施例５のレジスタ群４と第２演算部７の接続を示す図である。その他の構成は実施例１等と同等である。

【００９３】実施例５では、第２演算部７の中に乗算器７００、加算器７０１の２つの演算器を含み、使用すべき演算器を選択可能としている。このため、選択回路４３、４４とコントロールレジスタが新設される。すなわち、予めコントロールレジスタのｓｅｌビットによって

使用すべき演算器を設定することにより、制御信号４Ｆに従って所期の演算を行うことができる。３以上の演算器が存在するときは複数のｓｅｌビット（すなわちｓｅｌフィールド）を設ければよい。

【００９４】以上、本実施例によれば、演算器の数が増えてもレジスタの数を増加させる必要がない。また、演算の選択をコントロールレジスタで行うため、演算の指定に例えば図２の命令コード中のＯＰを使用する必要はなく、命令コード長のコンパクト化が可能となる。また、従来ならば命令によって指定されるべき演算をコントロールレジスタで指定するため、将来の拡張性を確保する意味でも好都合である。これらの結果、従来の課題３～５を解決することができる。

【００９５】実施例６．図１２は実施例６のレジスタ群４と第２演算部７の接続を示す図である。実施例６では、第２演算部７の中に乗算器７０２、加算器７０３の２つの演算器を含み、いずれかの演算器を単独で使用する場合、さらには乗算器と加算器を連結して使用する場合を選択できるよう、選択回路４６～４８と制御回路４５を備える。予め制御回路４５に、これら３通りの演算のうち所望の演算を設定すると、制御信号４Ｇによって選択回路４６～４８が制御される。

【００９６】１．乗算のみが選択されたとき

ＳＲ０、１へのデータ転送命令の実行に応じて乗算が実行され、選択回路４８で乗算器７０２の出力が選択され、ＳＲ２に乗算結果が書き込まれる。選択回路４６、４７の状態は任意である。

```
        MOV   SR2, 0
L1:     LD    SR0, (R0++)
        LD    SR1, (R1++)
        CB    R3, L1
```

従来のＲＩＳＣプロセッサで同様の処理を実行する場合、加算（ＡＤＤ）、乗算（ＭＵＬ）などの演算命令が不可欠であるが、この例では不要である。従って、本実施例でも従来の課題１、２を解決することができる。

【００９７】２．加算のみが選択されたとき

ＳＲ０、１へのデータ転送命令の実行に応じ、選択回路４６でＳＲ１側が選択され、選択回路４７でＳＲ０側が選択され、加算が実行される。選択回路４８では加算器７０３の出力が選択され、ＳＲ２に加算結果が書き込まれる。

【００９８】３．積和演算が選択されたとき

予め、ＳＲ２にゼロをロードしてクリアし、選択回路４６でＳＲ２側、選択回路４７で乗算器７０２の出力側を選択しておく。ここでＳＲ０、１へデータ転送命令の実行すると、まず乗算器７０２による乗算が実行され、その結果がＳＲ２に書き込まれる。この乗算結果はＳＲ２から選択回路４６を経て加算器７０３に入力され、一方、乗算結果は選択回路４７を経て加算器７０３へ入力される。この結果、ＳＲ０、１の乗算結果とＳＲ２の間で加算が行われる。この後、選択回路４８で加算器７０３の出力が選択され、ＳＲ２に加算結果が書き込まれる。

【００９９】こうした制御を繰り返し行うことにより、ＳＲ０、１へのロード命令を連続的に実行するだけで積和演算処理を行うことができる。以下、このプログラム例を示す。この例は、メモリ上にそれぞれＲ０、１をポインタとして格納されている２つの１次元配列の各要素の積をとり、それらの総和をＳＲ２に得る処理をループ処理によって記述したものである。

【０１００】

```
( SR2  ←0 )
( SR0  ← men(R0), R0  ← R0+1 )
( SR1  ← men(R1), R1  ← R1+1 )
(R３がゼロでない時Ｌ１に分岐)
```

演算処理が起動されるため、少ない命令サイクルでデータ処理を行うことができ、性能向上とプログラムサイズの縮小が可能となる。また、演算結果データの読み出しまで後続の演算結果データの書き込みが待たされるため、プログラミングまたはコンパイラ設計が容易となる。さらに本発明では、演算機能が追加、拡張されても、それが通常のデータ転送命令で起動されるため、別途演算命令を設ける必要がない。従って、命令コードサイズの長大化、ハードウェア規模の増大を抑制することができる。

【０１０４】別の態様では、命令の実行が演算結果データの読み出しを伴うとき、所望の演算結果データが出力用レジスタに書き込まれるまで読み出しが待たされるため、やはりプログラミング等が容易となる。

【０１０５】さらにこのとき、前記書き込みが待たされている間に入力用レジスタに対するデータ転送が行われた場合、第２演算手段による演算処理の起動が待たされるため、同様の目的が達成される。

【０１０６】入力用レジスタに有効なデータが保持され

【０１０１】以上、各実施例で本発明のデータ処理装置を説明した。ここでは、第２演算部７が備える演算器として、乗算器、加算器など２入力のものを挙げたが、これを例えば逆数演算、三角関数、対数関数などの１入力の演算器に置き換え、１つの入力用レジスタへの転送命令に応じて演算を起動するよう構成してもよい。

【０１０２】また各実施例では、汎用の作業用レジスタ（Ｒ０～３）と、第２演算部７による使用も可能なレジスタ（ＳＲ０～５）を同一の機能ブロック内にあるものとして図示したが、例えば本発明をＬＳＩ化する際、Ｒ０～３を演算部５と同じ回路ブロックに、ＳＲ０～５を第２演算部７と同じ回路ブロックに配置してもよい。

【０１０３】

【発明の効果】本発明のデータ処理装置によれば、データ転送命令の実行によって第２演算手段における所定の

ているとき、後続のデータ転送命令の実行による該レジスタへのデータ転送が待たされる場合も同様である。

【０１０７】また本発明では、入力用レジスタ及び出力用レジスタのレジスタセットが複数存在するため、用途に応じて別のレジスタセットを使用することができ、例えば割込み処理等を円滑に行うことができる。

【０１０８】第２演算手段が複数の演算器を有し、演算の選択が制御レジスタによって行われる場合は、選択内容を命令コード中に反映する必要がないため、命令コードサイズの短縮、将来の機能拡張に対する柔軟な対応が可能となる。

【０１０９】この場合さらに、ある演算器の演算結果データを別の演算器へ入力することによって複数の演算を連続的に実施することができる。この結果、例えば積和計算など連続的に高速実行することができる。

【０１１０】一方、入力用レジスタが複数存在するときは、１回の演算処理に必要な複数のデータが入力用レジスタに揃うまで演算処理の起動が待たされるため、プログラミングが容易となる。また、データの同時投入のためのハードウエアも不要であり、ハードウエア規模の縮小が可能となる。

【０１１１】入力用レジスタと第２演算手段との間にＦＩＦＯメモリが介挿される場合は、演算結果データの読み出し等が遅れた場合でもデータの入力を待たせる必要がなく、処理の高速化が可能となる。

【０１１２】以上、本発明によれば従来の課題をすべて解決することができる。

【図面の簡単な説明】

【図１】　ＲＩＳＣ方式のプロセッサの構成図である。

【図２】　従来のＤＳＰの特定命令のフィールド構成図である。

【図３】　実施例１に係るデータ処理装置の全体構成図である。

【図４】　図３のレジスタ群４と第２演算部７の接続を示す図である。

【図５】　実施例１の第２演算部７の内部構成図である。

【図６】　実施例２の第２演算部７の内部構成図である。

【図７】　図６の制御回路７２の内部構成図である。

【図８】　実施例３に係るデータ処理装置の全体構成図である。

【図９】　図８のレジスタ群４と第２演算部７の接続を示す図である。

【図１０】　実施例４のレジスタ群４と第２演算部７の接続を示す図である。

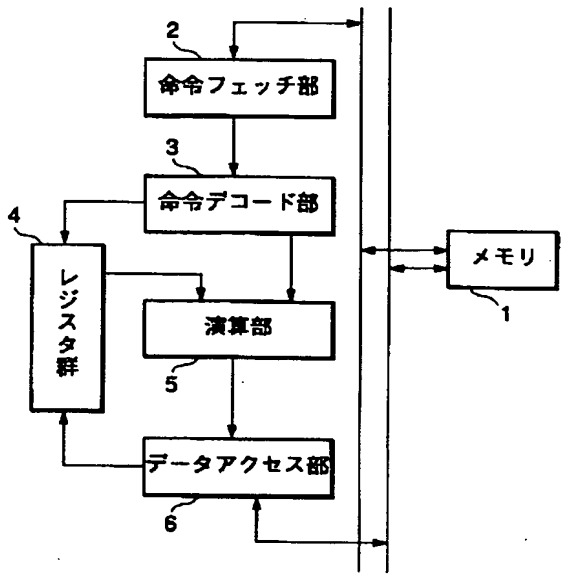
【図１１】　実施例５のレジスタ群４と第２演算部７の接続を示す図である。

【図１２】　実施例６のレジスタ群４と第２演算部７の接続を示す図である。

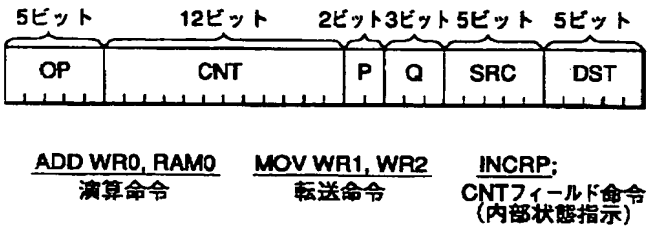
【符号の説明】

１　メモリ、２　命令フェッチ部、３　命令デコード部、４　レジスタ群、５演算部、６　データアクセス部、７　第２演算部、４１～４４，４６～４８選択回路、４５　制御回路、７１　演算回路、７２　制御回路、７３，７４　ＦＩＦＯ、７００，７０２　乗算器、７０１，７０３　加算器、７２４　計数回路Ａ、７２５　入力制御回路Ａ、７２７　計数回路Ｂ、７２８　入力制御回路Ｂ、７３０　演算制御回路。

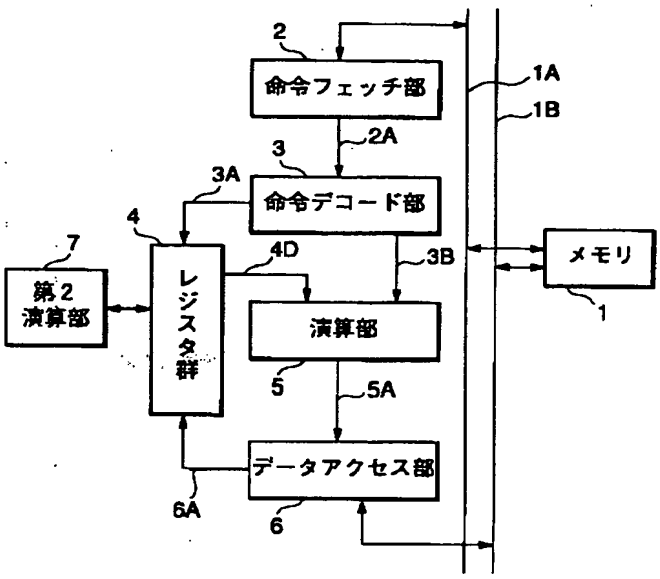
【図１】



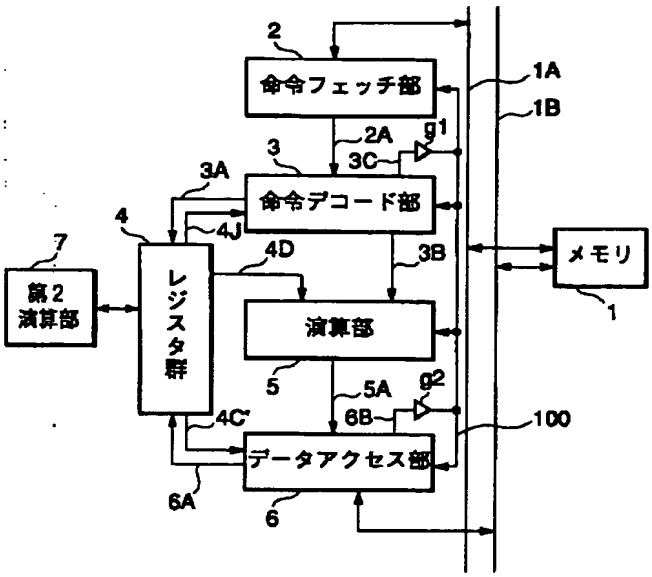
【図２】



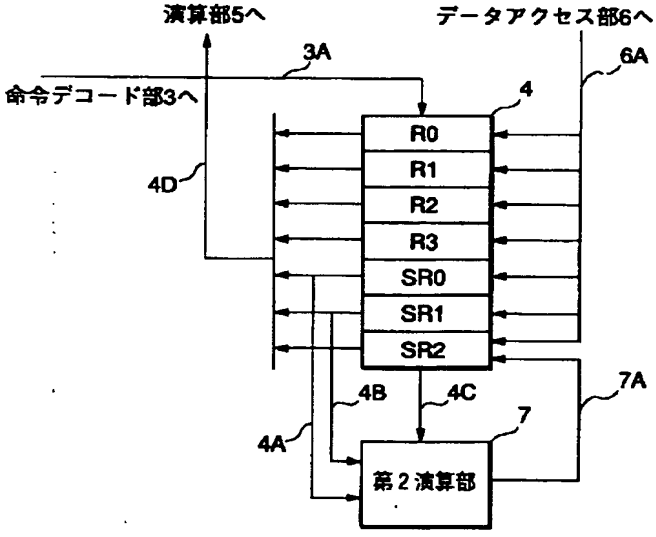
【図３】


2
命令フェッチ部
1A
1B
2A
3
3A
命令デコード部
4
7
第２演算部
レジスタ群
4D
3B
メモリ
演算部
1
5
5A
データアクセス部
6A
6

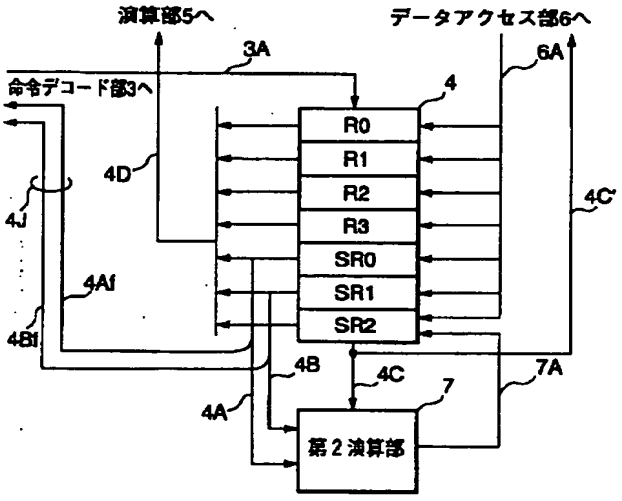
【図８】


2
命令フェッチ部
1A
1B
2A
g1
3
3C
3A
命令デコード部
4
4J
7
第２演算部
レジスタ群
4D
3B
メモリ
演算部
1
5
5A
g2
6B
100
4C'
データアクセス部
6A
6

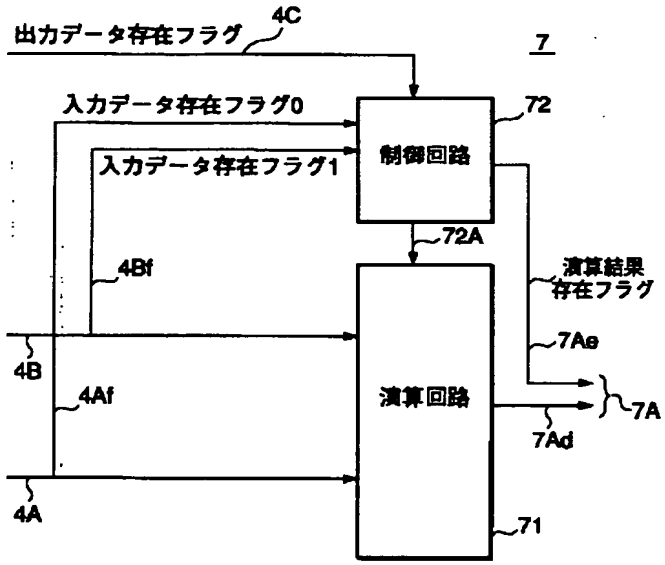
【図４】


演算部5へ
データアクセス部6へ
3A
6A
命令デコード部3へ
4
4D
R0
R1
R2
R3
SR0
SR1
SR2
4B
4C
7A
4A
7
第２演算部

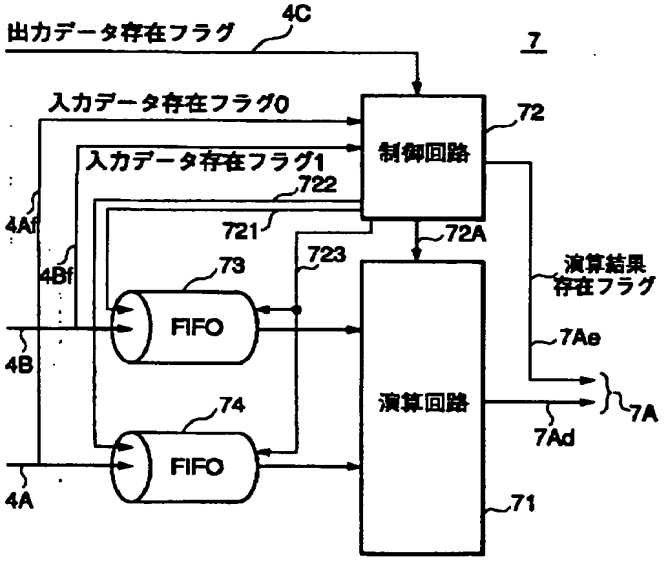
【図９】


演算部5へ
データアクセス部6へ
3A
6A
命令デコード部3へ
4
4D
R0
R1
R2
R3
SR0
SR1
SR2
4C'
4J
4A1
4B1
4B
4C
7A
4A
7
第２演算部

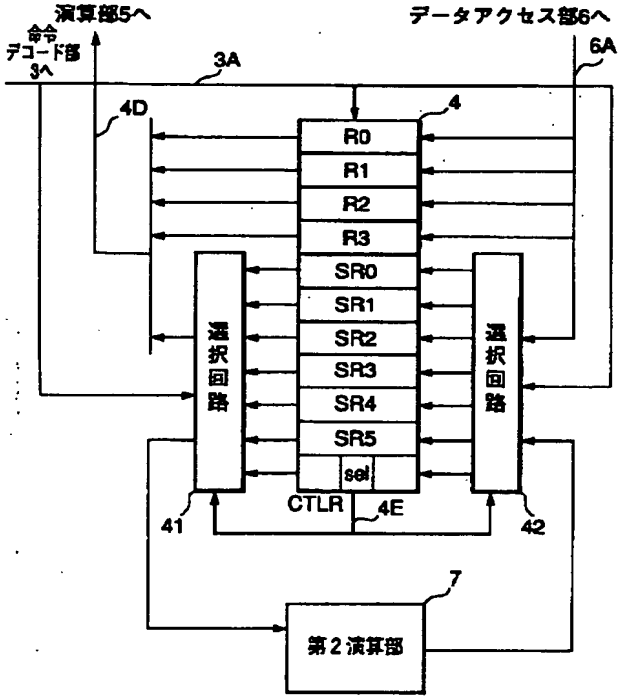
【図５】


出力データ存在フラグ
4C
7
入力データ存在フラグ0
入力データ存在フラグ1
制御回路
72
72A
4Bf
演算結果存在フラグ
7Ae
4B
4Af
演算回路
7A
7Ad
4A
71

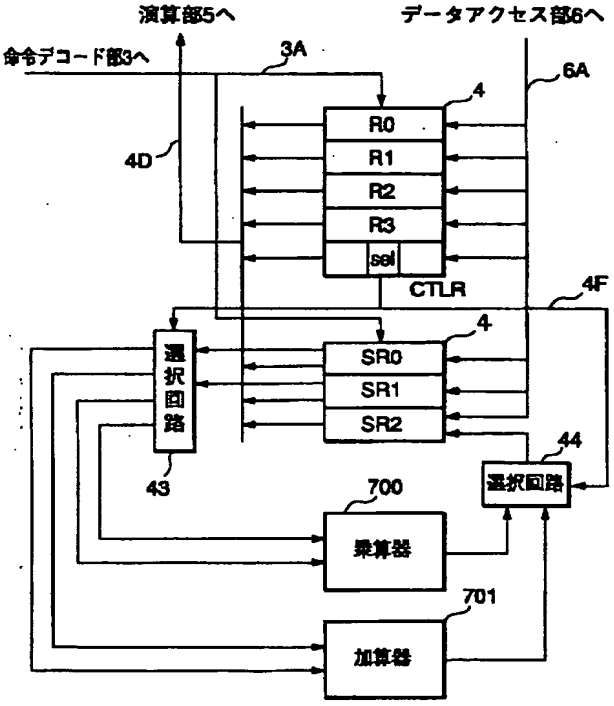
【図６】


出力データ存在フラグ
4C
7
入力データ存在フラグ0
入力データ存在フラグ1
制御回路
72
722
721
72A
4Af
4Bf
723
73
演算結果存在フラグ
FIFO
7Ae
4B
演算回路
7A
74
7Ad
FIFO
4A
71

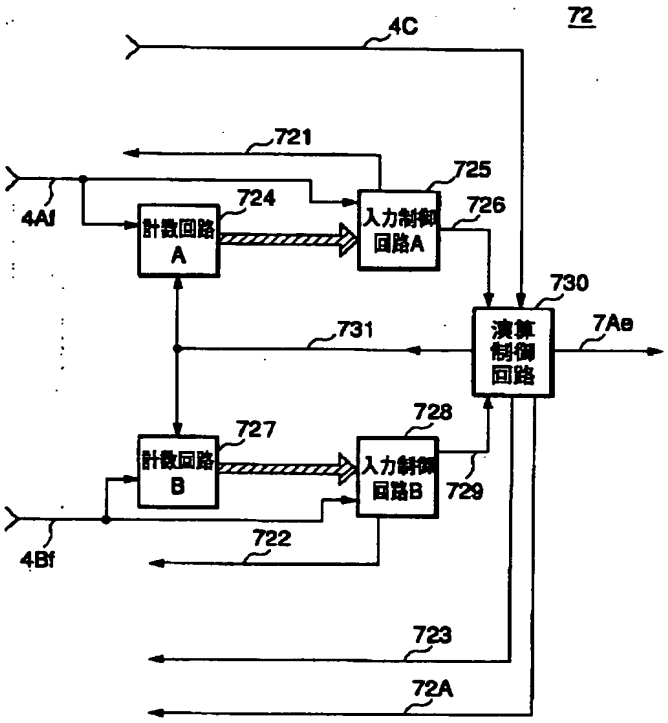
【図１０】


演算部5へ
データアクセス部6へ
命令デコード部3へ
3A
6A
4D
4
R0
R1
R2
R3
SR0
SR1
SR2
SR3
SR4
SR5
sel
選択回路
選択回路
CTLR
4E
41
42
7
第２演算部

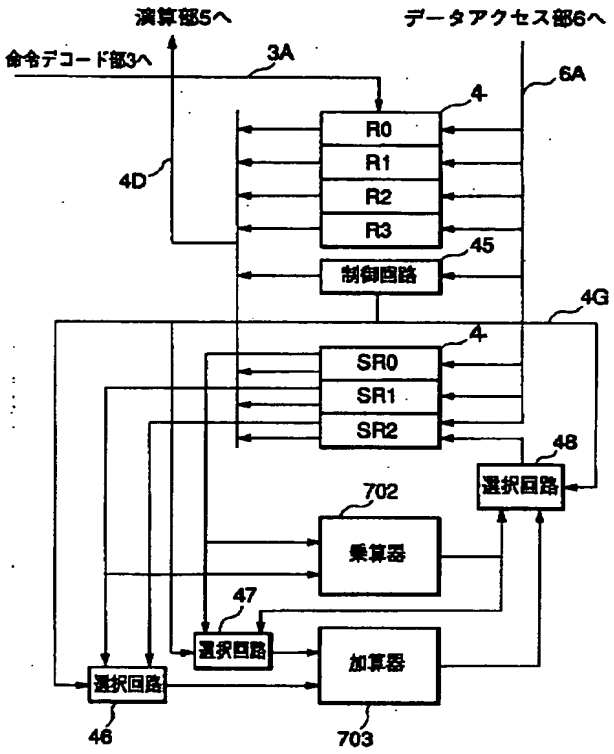
【図１１】


演算部5へ
データアクセス部6へ
命令デコード部3へ
3A
6A
4
4D
R0
R1
R2
R3
sel
CTLR
4F
4
選択回路
SR0
SR1
SR2
44
43
選択回路
700
乗算器
701
加算器

【図７】


72
4C
721
725
724
4AI
計数回路 A
入力制御回路A
726
730
731
7Ae
演算制御回路
727
728
計数回路 B
入力制御回路B
729
4Bf
722
723
72A

【図１２】


演算部5へ
データアクセス部6へ
命令デコード部3へ
3A
6A
4
R0
R1
R2
R3
4D
45
制御回路
4G
SR0
SR1
SR2
48
選択回路
702
乗算器
47
選択回路
加算器
選択回路
46
703